# APPENDIX B

## Integrating Sybase Adaptive Server Enterprise (ASE) with SnapMirror to Provide Disaster Recover

NetworkAppliance®
The evolution of storage.™

# Integrating Sybase Adaptive Server Enterprise (ASE) with SnapMirror® to Provide Disaster Recovery

Sridhara Gangoor, Network Appliance,
March, 2003 | TR 3242

TECHNICAL REPORT

Network Appliance, a pioneer and industry leader in data storage technology, helps organizations understand and meet complex technical challenges with advanced storage solutions and global data management strategies.

**Abstract**

Network Appliance, Inc. (NetApp), developed the Snapshot™, SnapRestore®, and SnapMirror features built into its operating system Data ONTAP™ to provide efficient and quick backup and recovery features. Data ONTAP leverages the WAFL® Snapshot capability to provide an automated file system replication facility called SnapMirror. Using SnapMirror technology, a filer can replicate one or more file systems to a partner filer, keeping the target file system synchronized with Snapshots that are created automatically on the source file system. This feature can be used to replicate the data to a different location to address any disaster recovery or to provide High Availability of a Sybase database environment. This paper describes how to integrate SnapMirror with Sybase database and application environments.



SYBASE®

# Table of Contents

---

# 1. Purpose and Scope

This document describes the steps necessary to set up a disaster recovery for Sybase Adaptive Server (ASE) for UNIX® using Network Appliance™ (NetApp®) SnapMirror technology. This paper discusses the following topics:

- Description of SnapMirror technology and its approach to disaster recovery
- The infrastructure requirement for using the SnapMirror solution
- The SnapMirror setup configuration procedure in the Sybase environment
- Disaster recovery in the field
- Resynching the replicated data following a recovery from a disaster
- Simulation of disaster recovery with a power failure
- Steps involved in SnapMirror setup

# 2. Assumptions

This paper describes the implementation of SnapMirror with Sybase ASE 12.5 in particular from an already functioning Solaris™ 8 and Network Appliance filer environment. We assume that you are familiar with Sybase ASE 12.5, as well as the operation of the filer, and you possess basic Solaris 8 administration skills. All examples in this technical report were tested using Sybase ASE 12.5 running under Solaris 2.8. The examples contained in this document may require modifications to run in your configuration. This document also assumes that you have access to the Sybase documentation for your particular operating system and that you have followed the documented steps for installation and configuration. Where Sybase documentation and this technical report conflict, you should assume that the Sybase documentation is correct. For more information on our SnapMirror product, please refer to our NetApp documentation: SnapMirror and SnapRestore: Advances in Snapshot Technology.

Please inform Network Appliance of any such contradictions so this document can be corrected.

- The name of the source filer is "filerA"
- The name of the target filer is "filerB"
- Sybase SA user is "sa" and password is not set
- All data is stored on a volume called `/vol/sybsource` on the source filer
- All data is replicated to a target volume called `/vol/sybtarget` on the target filer
- The name of the ASE server machine is "mondial"
- The name of the target ASE machine is "venus"
- ASE installed on source machine directory is `/sybase/ase125`
- ASE device files are located in `/sybase/data`
- ASE installed on the target machine is similar to source parameters

In our test setup `filerA-e4:/vol/syb` is a Gigabit Ethernet network connection used as NFS mount point.

## 3. Description of SnapMirror Technology

Network Appliance SnapMirror technology provides a data replication capability between filer volumes. Data on the source volume can be configured to replicate all the data periodically to a target at a user-defined time interval. The minimum range of definable time is one minute and the maximum is a month. At the end of each replication event, the source volume data is copied onto the target volume to the time it replicated the data. After this first replication, only the changed blocks are replicated to the target volume, minimizing the network traffic.

SnapMirror is an asynchronous mirroring product from Network Appliance, Inc. It allows you to replicate a volume on a source filer to another volume on a destination filer across either a LAN or a WAN connection. Once the baseline (level 0) transfer is complete the destination volume is accessible as a read-only copy. This allows remote data access as well as maintains a disaster recovery copy of the source volume's data. Only consistent file system images get transferred.

Systems fail, natural disasters occur, and mistakes happen. Data can be lost in any of these scenarios. NetApp filers integrate naturally with a variety of data protection and recovery methods. Users can choose an easy-to-use Snapshot utility for online backups, SnapMirror software for automated replication, SnapRestore software for quick restores, and SnapVault™ for centralized online backup of heterogeneous enterprise storage. Users can also select one of a variety of tape SAN and third-party NDMP-compliant tape backup solutions. Regardless of the method or configuration selected, NetApp filers provide optimized throughput and performance for all data protection operations.

The SnapMirror feature replicates data from one volume (the source) to another volume (the mirror) and periodically updates the mirror to reflect incremental changes on the source. The result of this

process is an online, read-only volume that contains the same data as the source volume at the time of the most recent update.

# 4. Infrastructures

The following are the requirements to set up a disaster recovery solution using SnapMirror in a Sybase environment with a NetApp filer:

- Sybase UNIX server machine running Sybase supported operating system (OS); this paper assumes Solaris 8 OS
- Source filer
- Target filer
- Supported network configuration for the above
- Sybase UNIX server machine for target location
- One or more NFS mount points with Gigabit Ethernet network connection
- For Fibre Channel Protocol, an FCP attachment kit is required from Network Appliance

## 4.1. Sybase Server Machine

### 4.1.1. Operating System

Sybase requires using the supported Operating System (OS) platform. If you are planning to use Storage Area Network (SAN) to support block protocol for the storage, Network Appliance requires checking the supported OS platforms. In our test case we used Sun Solaris 8 with Sybase ASE 12.5.0.3. We strongly suggest to installing the latest patches available from the OS vendors to address any fixes on the OS.

### 4.1.2. Hardware Configuration

*Server*

If you are using UFS, use Sybase supported OS versions for the disaster recovery configuration. However, if you are using the Direct Access File System (DAFS) protocol, you will be accessing the storage using raw devices known as DAFS Database Accelerator (DDA) devices.

It is important to realize that in order to utilize DDA, you must use a server that has the capability to support the appropriate VI/NIC cards that support DDA. This requirement will be met if you use Sun™ Ultra-60 or higher models as your server. You can also set up the disaster recovery configuration using the NetApp SAN solution.

*Memory*

It is recommended to configure these parameters in the /etc/system file based on the system resources available.

*Network*

In addition to a regular network connection, a Gigabit Ethernet connection is required. In a DAFS test environment, a DAFS connection can be provided through Emulex Virtual Interfaces (VI) NICs that are

direct-connection LC-to-LC multimode fiber-optic cable. Multiple VI NIC paths between the filer and the Sun machine increase the performance. The following diagram gives an idea about the network configuration.

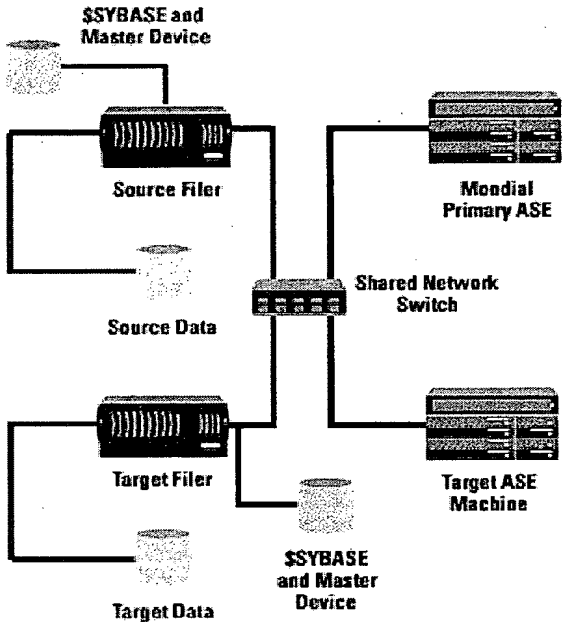


**Figure 1.** System Configuration Example

### 4.1.3. /etc/system File

This section is useful only for Sybase ASE configuration and not applicable to SnapMirror configuration. Some examples for the configuration values (/etc/system) on the Sybase Solaris server machine are provided below. Please keep in mind that the setting of all parameters below is completely optional, with the exception of the shared memory parameters, which can be configured based upon available resources.

```
set shmsys:shminfo_shmmax=8589934592
set shmsys:shminfo_shmmin=1
set shmsys:shminfo_shmmni=900
set shmsys:shminfo_shmseg=300

set semsys:seminfo_semmap=600
set semsys:seminfo_semmni=1000
set semsys:seminfo_semmns=1400
set semsys:seminfo_semmnu=800
set semsys:seminfo_semume=400
set semsys:seminfo_semmsl=1400
set semsys:seminfo_semopm=400

*--------------------
```

```
* Message Queue
*--------------------
set msgsys:msginfo_msgmap=1024
set msgsys:msginfo_msgmax=65535
set msgsys:msginfo_msgmnb=65535
set msgsys:msginfo_msgmni=1024
set msgsys:msginfo_msgssz=2048
set msgsys:msginfo_msgtql=1024

*
*Increases the size of STREAMS synchronization
set sq_max_size = 1600
set nstrpush = 90
*
set ncsize 8000
set maxusers = 2048
set nfs:nfs3_max_threads = 48
set nfs:nfs3_nra = 10
set priority_paging=1
```

If your configuration requires a larger value for kernel stack space, you may use the following example. Configuration of the parameters below in the "`/etc/system`" file will increase the kernel stack space to 16KB.

```
set lwp_default_stksize=0x4000
set rpcmod:svc_run_stksize=0x4000
```

Newer NearStore products will automatically configure the RAID groups for a volume and properly define the RAID groups across NearStore shelves. Please check with your service provider to see whether your NearStore has this new feature.

## 4.2. NetApp Source Filer

The NFS and SnapMirror licenses on both the source and the target filers are to be enabled. Source volume for putting Sybase data and system files is required on the source filer. It is suggested to avoid filling up of the entire source volume. More information on integrating Sybase ASE with the filer can be obtained by visiting our technical library.

## 4.3. Target Filer

As mentioned earlier, checks for necessary licenses are activated on the target filer. The target volume must be created, and the size of the target volume should not be less than the size of the source volume.

## 4.4. Network

The network connection between the ASE server machine and the filer must be established. Similarly, the network connection must be established for the target ASE machine and the target filer. Dedicated/private network connections between the Sybase server machine and the NetApp filer are recommended for the following reasons:

- Any issues of contention or latency are eliminated if the Sybase server machine and the NetApp filer are the only two nodes in the network

- Creating a private network connection ensures security; there is no issue of protecting the Sybase data files from tampering, as would be the case on a shared network
- If additional NetApp filers are added, they receive the same security as the initial NetApp filer
- The use of Jumbo Frames can be easily implemented
- Broadcast storms and other performance-robbing network anomalies are eliminated from the database configuration

## 4.5. Filer NFS Mount Points

Unless you are using our FCP SAN product, NFS mount point on UNIX OS is required. This will allow configuration of file permissions associated with existing files on the filer volume. Some of the nondefault mount options to be set between the filer and the Sun machine are:

```
-o hard, intr, vers=3, proto=tcp are required. Note that if you set the
protocol to udp, the Sybase server might encounter some issues.
```

The authorization for each user who will have access to each NFS mount point can be configured and typically it is set to rwx. It is recommended you ensure that each filer volume on the filer has UNIX or mixed security style. Using the following command syntax can do this:

```
qtree security [vol_name] [unix | ntfs | mixed]
```

where vol_name is the name of the filer volume for which the security style is set to UNIX or mixed style.

For example, the following command would be used to set the security style for a volume `/vol/sybase` to UNIX:

```
qtree security /vol/sybase unix
```

Similarly, a security for qtree can be set by using:

```
qtree security [qtree_path] [security style]
```

## 4.6. Setting Sybase Volume Security Styles

You must also ensure that the volume(s) to be used for Sybase storage on the filer has either UNIX or mixed security style by using the qtree security command on the appropriate volume(s) as indicated below:

```
qtree security /vol/{volname} [ unix | mixed ]
```

## 4.7. Sybase User Account

Sybase recommends that you have a username on the Solaris system called "sybase." Once this user account has been set up, make sure it has the appropriate permissions on the Solaris system. For further details regarding this procedure, please refer to the Sybase installation guide.

### 4.8. Sybase Pre-installation Requirem nts

Prior to beginning the Sybase ASE 12.5 installation process, make sure that you complete the preinstallation tasks as indicated in the Sybase ASE installation guide. This includes selection of the location where the Sybase binaries will be installed as well as selection of the ASE server name. Note that Sybase requires the TCP/IP protocol for communication between the ASE server and its client. For further details on the Sybase requirements, please refer to the Sybase ASE installation guide.

## 5. Setting up SnapMirror for Sybase ASE Database

The basic steps are to:

- Identify the source and target volumes
- Estimate data change rates
- Edit configuration files
- Start and stop the SnapMirror process

To use SnapMirror, create a volume to be used as the destination (mirror) for data replication. The mirror can reside on the same filer as the target volume or on a different filer. If the target volume is configured on the same filer and the filer is lost due to a disaster, the mirrored data might be lost.

To replicate data for the first time, the filer transfers all data in all Snapshots in the source volume in addition to the data itself on the volume/qtree to the target location. After the filer finishes transferring the data, it brings the mirror online, but in a read-only state. This version of the mirror is the baseline for future incremental changes.

### 5.1. Identify the Source and Target Volumes

Identify the volume/qtree from which the data is to be replicated to the target volume. In our test environment, the source volume is `filerB:/vol/sybsource` and the target is `filerA:/vol/sybtarget`, where filerB is the source filer and filerA is the target filer.

### 5.2. Calculate the Rate of Change of Data

Estimate how much the data changes at the source site. This allows us to configure the mirroring interval properly and enable using the network efficiently.

### 5.3. SnapMirror Configuration Files

#### 5.3.1. Control Files

There are three files to be edited on all filers involved with SnapMirror, and they are:

- /etc/rc,
- /etc/snapmirror.conf[1]
- /etc/snapmirror.allow

### 5.3.2. Configuration Parameters

On both the source and the target filer, append the `/etc/rc` file with "vol snapmirror on" without double quotes. Edit `/etc/snapmirror.conf` on the source filer and add the configuration parameters similar to the following lines. Please replace the parameters according to your environment. A simple configuration is shown below. The last line is commented out with different configuration parameter settings.

```
filerA:sybsource filerB:sybtarget - 2 * * *
#sourcefiler:sourcevolume targetfiler:targetvolume networkthrottle
minute hour dayofmonth dayofweek
#filer1:svolume filer2:tvolume kdb=7500 0,10,20,30,40,50 1,7,13,19,23 *
1-5
```

In the above example, the first line is configured to SnapMirror the data on the source volume called sybsource residing on the source filer called filerA to the target volume sybtarget on the target filer called filerB with available network throttle for every two minutes of every hour of each day of the month on every day of the week. The "-" option is not restricting the network throttle, 2 being the SnapMirror frequency and * * * represents hour, day of the month, and day of the week. Care must be taken while setting up the mirroring interval. It is recommended to calculate the rate of change of data for each SnapMirror interval. If the changes cannot be mirrored before the start of the next SnapMirror, it will be reverted back and new mirroring starts.

The last line in the above example is commented out, and it represents a customized SnapMirror configuration. On both filer volumes edit another file `/etc/snapmirror.allow` and add both source and target filers. If you are mirroring between multiple filers, you can enter the names of all the filers involved in mirroring. In our test setup, the following was the entry of our `/etc/snapmirror.allow` file:

filerB
filerA

The above `snapmirror.conf` file indicates that the two filers involved in the mirror are filerB and filerA. filerA is the source filer and filerB is the target filer. Both filers have identical `/etc/snapmirror.allow` files.

The data on the sybsource volume will be replicated onto the target volume called sybtarget. In our setup, we did not set any maximum network throttle on the mirror. Snapmirror.allow files can have different entries on different filers depending on the SnapMirror configuration.

The SnapMirror interval is set to occur once in two minutes, every hour, every day of the month, and every day of the week. It is recommended to set this parameter carefully according to the need of data replication at your site. Setting SnapMirror for every minute makes the network traffic busy and it may get rolled back before the next replication event starts. In that scenario, use "snapmirror update" command to force the SnapMirror update to the target site.

"vol snapmirror on" in the `/etc/rc` file enables the SnapMirror process.

After the initial setup, use the "snapmirror on" command. The "snapmirror initiate" command initiates the replication event, and once the base-level replication is done, SnapMirror replicates only the changed block. After the replication phase the target volume will reflect a status of "online snapmirrored" when queried with the "vol status" command. The target volume is now online in read-

only mode. The source volume continues to be online and available for applications at any stage of the data-replication operation.

## 6. SnapMirror Software Operation

The SnapMirror feature replicates data from one volume (the source) to another volume (the mirror) and periodically updates the mirror to reflect incremental changes on the source. The result of this process is an online, read-only volume that contains the same data as the source volume at the time of the most recent update.

To use SnapMirror, create a volume (if it's not already created) to be used as the destination (mirror) for data replication. The mirror can reside on the same filer as the source volume or on another filer. To make incremental changes on the mirror, the filer takes regular Snapshots on the source volume according to the schedule specified in the /etc/snapmirror.conf file. By comparing the current Snapshot with the previous Snapshot, the filer determines what changes it needs to make to synchronize the data in the source volume and the data in the mirror. For example, if a file is created or deleted from the source volume, SnapMirror creates or deletes the corresponding file in the mirror the next time it updates the mirror. SnapMirror automatically deletes old Snapshots that are no longer necessary for data replication.

SnapMirror has two phases: initialization for the first time and then the incremental update. The initialization phase consists of a level 0 replication event in which a Snapshot is created on the source volume and then sent in its entirety to the target volume. For example, a 250GB mirror initialization takes more than an hour to complete across a Gigabit Ethernet network. The level 0 event serves to initialize or "seed" the mirror volume since it contains every block in the source volume as of the point in time when the Snapshot is created.

After mirror initialization is complete, the filer examines /etc/snapmirror.conf every minute (depending upon the parameter set) to see if there are any scheduled updates. This allows for modification of the mirror's configuration without disrupting the mirror. When an incremental update schedule time is due, a new Snapshot is taken and compared to the previous Snapshot. The delta blocks and the block map file are sent to the mirror target. In contrast to the level 0 initialization, the data mirrored is typically much smaller. Note that at all times the mirror target file system is in a consistent state.

After the required files are edited, the SnapMirror process can be started in this sequence: Put the target volume in offline mode. This makes sure that no user is using that volume. The volume can be put in offline mode by issuing the command "vol offline targetvolume" on the target filer console, where targetvolume is the name of the target volume where the data will be replicated from the source volume.

Start the SnapMirror initialization by entering "snapmirror initialize -S sourcefiler:sourcevolume targetfiler:targetvolume" on the target filer. For detailed syntax, refer to the Data ONTAP System Administrator's Guide.

Using the "snapmirror status" command, check the status and if the mirroring has not started, read the output of earlier command; maybe the target volume is not offline or needs to download the bootblock to the target volume disks added. In that case, bring the volume online and allow completion of the bootblock. Then turn it offline and start the SnapMirror initialization process. When a different geometry of disks is used, download of the bootblock may be required, which will be done automatically.

Some of the points to be noted here are:

- During the mirroring, the target volume is not accessible
- If an initialization replication event is interrupted, such as a network outage for a longer time, the SnapMirror process aborts and revert back to a previous consistent state; this means partial replication is not allowed and the SnapMirror process has to be restarted
- The maximum throttle (kilobytes per second) parameter in `/etc/snapmirror.conf` can be changed at any time, and the new values take effect within two minutes except in the case of bandwidth throttle already in effect; in such cases the changes take effect once the initialization is over or the process is interrupted
- Incremental updates start only after the level 0 initialization
- If the changes could not be replicated before the start of another incremental update, changes are reverted back and the new replication update process starts; this means subsequent incremental updates transmit the data, which are missed, so no data loss can occur
- The SnapMirror process can be stopped and restarted at any time by issuing the following command sequence on either filer:

  ```
  snapmirror off
  [time interval]
  snapmirror on
  ```

- As long as the target volume remains read-only during the time between turning the SnapMirror process off and on, the mirror maintains its atomicity status
- To break the mirror, issue the following command on the target filer: "snapmirror break targetvolume" where targetvolume is the name of the target volume; this makes the target volume to be read/write accessible changed from read-only mode.

# 7. Our SnapMirror Setup

As a test scenario, we installed Sybase ASE 12.5.0.3 on the Solaris 8 server and the $SYBASE uses the filer volume.

## 7.1. Configuration Details

- ASE Server Solaris server name: `mondial`
- Source filer name: `filerA`
- filerA-e10:`/vol/sybsource hard,intr,proto=tcp,vers=3,rsize=32768,wsize=32768` on `/Sybase` where filerA-e10 is a Gigabit Ethernet network
- Sybase home directory $SYBASE: `/sybase/ase125`
- Database created is `userdb1` on `userdev1,userdev2` and log on `logdev1` devices

- Tables created are: `title`, `titlea`, `titleb` and `titlec`
- Table created during the replication is: `class`

## 7.2. Filer Volumes

Now we set up the target filer volumes. We selected "filerB" as our target filer. Since the target filer is a different filer, we created a volume called `/vol/sybtarget` and mounted it with the following command on a different Solaris 8 server called "venus."

### 7.2.1. Mount Options

# mount -o hard,intr,vers=3,proto=tcp,rsize=32768,wsize=32768 filerB-e4:/vol/sybtarget /sybase where filerB-e4 is a Gigabit Ethernet path.

Note that we used the same mount options on the source ASE server machine "mondial" (Solaris 8 server).

### 7.2.2. Configuration Files

Now check the SnapMirror configuration required files. Let's first check the /etc/rc file on the source filer.

**/etc/rc on the source filer**

```
# cat /svol0/etc/rc
#Auto-generated by setup Mon Feb 10 16:37:11 PST 2003
routed on
options dns.enable off
options nis.enable off
savecore
exportfs -a
nfs on
vol snapmirror on
```

**/etc/rc on the target filer**

Next we checked out `/etc/rc` file on the target filer "filerB"

```
# cat /tvol0/etc/rc
#Auto-generated by setup Tue Feb 11 12:30:15 PST 2003
routed on
options dns.enable off
options nis.enable off
savecore
exportfs -a
nfs on
vol snapmirror on
```

**/etc/snapmirror.conf**

The next step is to edit the `/etc/snapmirror.conf` file on both filers. We included this file on both filers even though it's not a requirement.

```
# cat /svol0/etc/snapmirror.conf
#filerA:sybsource filerB:sybtarget -** * *
#filerA:sybsource filerB:sybtarget - 11,21,31,41,51 8,9,10,11,12,
   13,14,15,16,17,18,19,20 * 1,2,3,4,5
#filerA:sybsource filerB:sybtarget - * * * *
```

**/etc/snapmirror.allow**

The last file to check/edit is /etc/snapmirror.allow. On our test setup these files on both filers are identical.

```
# cat /svol0/etc/snapmirror.allow
filerA
filerB
```

### 7.2.3. Selecting other Parameters

At this time, we have configured the required files such as `/etc/rc`, `/etc/snapmirror.conf`, and `/etc/snapmirror.allow` on both filers. If you are planning not to burden the network or slow down the replication event, you can specify the maximum network usage throttle to be set. Note that this value is purely arbitrary, and it is suggested you calculate the right amount of maximum network throttle suitable for your environment.

In our `snapmirror.conf` configuration file, we have set replication at every one minute. A simple configuration example is shown in the next line, which is commented out.

```
#filerA:sybsource filerB:sybtarget - * * * *
```

Our snapmirror.allow files are identical on both filers. It is okay to add other filer names if the SnapMirror process is configured for mirroring data between filers. Initial configurations do not come with preconfigured `snapmirror.conf` or `snapmirror.allow` files, and these files are to be edited with appropriate entries.

As we mentioned earlier, on both source and target filers, enter the following command.

```
snapmirror on
```

## 7.3. SnapMirror Available Volumes

It is important to note that:

- If SnapMirror is to be used as part of the backup and recovery scenario, it is important to note that SnapMirror is only available to nonroot volumes. The source and target volumes may have different sizes or different disk geometries. However, the target volume should be the same size or larger than the source volume. Maintaining the geometry is known to yield better performance compared to different geometry configurations.
- Once the target volume is selected, make sure it's the same or larger than the source volume. On our test environment, after creating the necessary volumes, we checked using the "vol status" command.

## 7.4. Volume/qtree Status

On our system, we checked the volume or qtree status to make sure they are available for replication.

```
filerB> vol status
Volume State        Status        Options
vol1 online         normal        nosnapdir=on, raidsize=14
vol0 online         normal        root, nosnapdir=on
sybtarget online    normal
filerB>
```

Similarly, we checked on the source volume.

```
filerA> vol status
```

```
Volume State          Status         Options
vol0 online           normal         root, nosnapdir=on
sybsource online      normal
```

## 7.5. Start of SnapMirror

Now we are ready to initiate the mirroring data using the "snapmirror" command. In our case, we issued the following command on the target volume and the output is given below:

```
filerB> snapmirror initialize -S 10.32.90.15:sybsource sybtarget
Transfer started.
Monitor progress with 'snapmirror status' or the snapmirror log.
filerB> Mon Mar  3 09:33:52 PST [worker_thread_161:notice]:
Automatically restricting destination volume sybtarget (currently offline)
```

### 7.5.1. SnapMirror Status

Once the mirroring is started, the progress can be monitored using the "snapmirror status" command.

```
filerA> snapmirror status
Snapmirror is on.
Source              Destination         State     Lag     Status
filerA:sybsource    filerB:sybtarget    Source    -       Transferring  (1600MB
done)
filerA>
filerA> snapmirror status
Snapmirror is on.
Source              Destination         State     Lag     Status
filerA:sybsource    filerB:sybtarget    Source    -       Transferring  (1800MB
done)
filerA>
```

### 7.5.2. Database Activity During Mirroring

While SnapMirror was in progress, we connected to the ASE server using the "isql" utility and checked the number of rows.

```
1> use userdb1
2> go
1> select count(*) from title
2> go
  -----------
       524288

(1 row affected)
```

After counting the number of records in the table called "title," we inserted another 524,288 rows. The output of this operation is shown below.

```
1> insert into title select * from titlea
2> go
(524288 rows affected)
1>
```

In the meanwhile, we checked the SnapMirror status. The output is given below:

```
filerB> snapmirror status
Snapmirror is on.
Source              Destination         State          Lag  Status
```

```
filerA:sybsource  filerB:sybtarget  Uninitialized  -    Transferring (2980 MB
done)
```

### 7.5.3. Create a Table on the Existing Database

During this process, we decided to create another table on the database "userdb1" called "class." The procedure used to create the new table is shown below.

```
1> create table class (number int, name varchar(36))
2> go
```

### 7.5.4. Insert Data into the Table Created

```
1> insert into class (number, name) values(123,
   "the snapmirror is easy to setup and use")
2> go
(1 row affected)
....
1> insert into class select * from class
go
(131072 rows affected)
```

## 7.6. Back Up Sybase Master Database Information

Since we are working on data mirroring and disaster recovery, it is suggested you note down the output of the following commands. For more details, refer to Sybase ASE product manuals.

```
1> select * from sysusages order by vstart
2> go
1> select * from sysdatabases
2> go
1> select * from sysdevices
2> go
```

## 7.7. Monitor SnapMirror Status

While the database server was active, we continue to monitor the progress of SnapMirror.

```
filerB> snapmirror status
Snapmirror is on.
Source                  Destination       State          Lag    Status
10.32.90.15:sybsource   filerB:sybtarget  Uninitialized  -      Transferring
    (33 GB done)
filerB> snapmirror status
Snapmirror is on.
Source                  Destination       State          Lag       Status
filerA:sybsource        filerB:sybtarget  Snapmirrored   00:56:23  Idle
filerB>
```

## 7.8. Database Activity

### 7.8.1. Altering the User Database

As a routine database activity, now we will alter the database userdb1 by 500MB on userdev1 and 150MB on logdev1, which should be replicated to the target location.

```
1> alter database userdb1 on userdev1=500 log on logdev1=150
2> go
```

```
Extending database by 128000 pages (500.0 megabytes) on disk userdev1
Extending database by 38400 pages (150.0 megabytes) on disk logdev1
```

### 7.8.2. Inserting Data into Existing Tables

```
1> insert into titlec select * from titlea
2> go
(524288 rows affected)
1> select count(*) from titlec
2> go
       524288
(1 row affected)
1>
```

### 7.8.3. Breaking the Mirror

If the mirroring is to be broken, login to the target filer console and use the "snapmirror break" command. On our test setup, the output is shown below when the mirror was broken.

```
filerB> snapmirror break sybtarget
snapmirror break: Destination sybtarget is now writable.
FilerB>
```

### 7.8.4. Resync the Mirror

The mirror could be broken for many reasons not restricted to reporting or development purposes. Note that all changes made to the target will be lost after the resync process. The target volume can again be resync'd with the source data. On our setup, the output is shown below when the resync command was issued:

```
filerB> vol online sybtarget
Mon Mar  3 12:44:06 PST [download.update:info]: Begin bootblock update,
   prototype is 8.0
Volume 'sybtarget' is now online.
filerB> snapmirror resync -S 10.32.90.15:sybsource sybtarget
The resync base snapshot will be: filerB(0033587173)_sybtarget.1
These newer snapshots will be deleted from the destination:
    hourly.0
Are you sure you want to resync the volume? y
Mon Mar  3 12:44:42 PST [rc:notice]: snapmirror: resynching volume sybtarget
to
   10.32.90.15:sybsource using filerB(0033587173)_sybtarget.1 as the base
snapshot.
Volume sybtarget will be briefly unavailable before coming back online.
Mon Mar  3 12:44:42 PST [config_async_0:notice]: CIFS - disabled for volume
sybtarget
Volume sybtarget is now restricted.
Mon Mar  3 12:44:43 PST [config_async_0:notice]: Reverting volume sybtarget to
a
  previous snapshot.
Mon Mar  3 12:44:43 PST [license.db.migrate.vol.readonly:warning]: Skipping
volume
  sybtarget migration because it is read-only
Volume 'sybtarget' is now online.
Volume sybtarget: revert successful.
Revert to resync base snapshot was successful.
Transfer started.
```

#### 7.8.4.1. Monitor SnapMirror Status

Monitor progress with 'snapmirror status' or the snapmirror log.

```
filerB>
filerB> snapmirror status
Snapmirror is on.
Source                  Destination       State          Lag        Status
10.32.90.15:sybsource   filerB:sybtarget  Snapmirrored   03:13:54
Transferring
   (688 MB done)
filerB> Mon Mar  3 12:46:55 PST [download.updateDone:info]:
   Bootblock update completed
filerB> snapmirror status
Snapmirror is on.
Source              Destination        State          Lag        Status
filerA:sybsource    filerB:sybtarget   Snapmirrored   00:08:26   Idle
filerB>
```

#### 7.8.4.2. Break the Mirror

On the target filer, break the mirror using the SnapMirror break command.

```
filerB> snapmirror break sybtarget
snapmirror break: Destination sybtarget is now writable.
filerB>
```

# 8. Disaster Recovery

## 8.1. Using the Same Environment on the Target

If you are planning to use the same configuration as the source site, ensure the target volume/qtree is mounted and the mirrored files have ownership to the Sybase account you intend to use. In our test setup, we mounted the mirrored data volume and changed the file ownership to "sybase" account user on the Solaris server machine "venus."

Once the SnapMirror break occurs, the target volume becomes writeable. To bring the Sybase database server onto the target OS machine, login to that machine. In our test setup, we decided to use the same environment as the source setup. Hence, we logged in as "sybase" user on the Solaris machine, which is a different machine from the source ASE server UNIX machine. In order to maintain the uniformity, we changed the file ownership to "sybase" account using the "chown" command. Now login as "sybase" user on the target. Change the file ownership to "sybase" account, including the device files. In our setup, we executed the following commands:

```
#chown sybase:others /sybase/* /sybase/*/*/
   /sybase/*/*/* /sybase/*/*/*/*
#chown sybase:others /sybase/. /sybase/.. /sybase/ase125/.
   /sybase/ase125/.. /sybase/data/. /sybase/..
```

### 8.1.1. $SYBASE/Interfaces File before Changes

After checking the file permissions, the next step is to edit the interfaces file. In our setup, we edited the interfaces file located on the ASE home directory ($SYBASE). The following example shows the contents of the $SYBASE/interfaces file before making changes as replicated from the source filer:

```
sri
```

```
        master tli tcp /dev/tcp \x000210040a0a5a0a0000000000000000
        query tli tcp /dev/tcp \x000210040a0a5a0a0000000000000000

sri_back
        master tli tcp /dev/tcp \x000210680a0a5a0a0000000000000000
        query tli tcp /dev/tcp \x000210680a0a5a0a0000000000000000

sri_mon
        master tli tcp /dev/tcp \x000210cc0a0a5a0a0000000000000000
        query tli tcp /dev/tcp \x000210cc0a0a5a0a0000000000000000

SRI_XP
        master tli tcp /dev/tcp \x000211300a0a5a0a0000000000000000
        query tli tcp /dev/tcp \x000211300a0a5a0a0000000000000000

sri_text
        master tli tcp /dev/tcp \x000211940a0a5a0a0000000000000000
# End of "interfaces" file
```

### 8.1.2. $SYBASE/Interfaces File after Changes

Since we are trying to configure the ASE server on a different machine, we need to edit the entries appropriately. To complete this task, note down the IP address of the ASE server machine (Solaris 8 machine in our case). Let's assume the IP address of the Solaris machine is 10.10.90.10.

In the entry \x000211300a0a5a0a0000000000000000, update entry of eight characters before 16 zeroes. For example, change 0a205aa0 (which represents 10.10.90.10) to 0a0a5a06 (which is 10.10.90.6). Make these changes to other Sybase servers such as backup and monitor servers, etc. The modified interfaces file on our replicated volume is listed below:

```
sri
        master tli tcp /dev/tcp \x000210040a0a5a060000000000000000
        query tli tcp /dev/tcp \x000210040a0a5a060000000000000000

sri_back
        master tli tcp /dev/tcp \x000210680a0a5a060000000000000000
        query tli tcp /dev/tcp \x000210680a0a5a060000000000000000

sri_mon
        master tli tcp /dev/tcp \x000210cc0a0a5a060000000000000000
        query tli tcp /dev/tcp \x000210cc0a0a5a060000000000000000

SRI_XP
        master tli tcp /dev/tcp \x000211300a0a5a060000000000000000
        query tli tcp /dev/tcp \x000211300a0a5a060000000000000000

sri_text
        master tli tcp /dev/tcp \x000211940a0a5a060000000000000000
```

```
query tli tcp /dev/tcp \x000211940a0a5a060000000000000000
```

Note that only the necessary network address information is changed in our setup. If you're planning to use a different port, you need to update correctly for the four digits starting after \x0002. If any application is already using these ports, the Sybase server will fail to start as it cannot start a listener service. For detailed information, please refer to Sybase ASE manuals.

### 8.1.3. Bringing up the Sybase ASE Server

Now we are ready to bring up the Sybase ASE server. we checked once again to make sure we are using the right path for starting ASE server on the target machine.

Make sure the filer volume is mounted properly on the target UNIX server.

```
venus% mount | grep sybase
/sybase on 10.32.90.22:/vol/sybtarget
remote/read/write/setuid/hard/intr/vers=3/
   proto=tcp/rsize=32768/wsize=32768/dev=3f00005 on Wed Jan 12 17:55:43 2000
```

### 8.1.4. Bringing up the Database at the Target Location

#### 8.1.4.1. RUN_<ASE server> script

After the file ownership, permission, and interfaces file entry updates, execute the Sybase environment setup file located under $SYBASE path. This will set up the necessary environmental variables to use Sybase ASE. To this file, we added the DSQUERY and DSLISTEN variables. On our test machine, we executed RUN_sri script located under $SYBASE/$SYBASE_ASE/install directory. The output is shown below:

```
venus% ./RUN_sri &
[1] 1508
venus% 00:00000:00000:2003/03/03 16:50:28.54 kernel  Use license file
/sybase/ase125/SYSAM-1_0/licenses/license.dat.
00:00000:00000:2003/03/03 16:50:28.54 kernel  Checked out license ASE_SERVER
00:00000:00000:2003/03/03 16:50:28.54 kernel  Adaptive Server Enterprise
Edition
00:00000:00000:2003/03/03 16:50:28.56 kernel  Using config area from primary
   master device.
00:00000:00000:2003/03/03 16:50:28.63 kernel  Using 1024 file descriptors.
00:00000:00000:2003/03/03 16:50:28.64 kernel  Adaptive Server
Enterprise/12.5.0.3/EBF
   10689 IR/P/Sun_svr4/OS 5.8/rel12503/1915/64-bit/FBO/Thu Jan 23 16:05:19
2003
...
00:00000:00002:2003/03/03 16:50:30.69 kernel  ninit:0: listener raw address:
   \x000210040a205a060000000000000000
00:00000:00002:2003/03/03 16:50:30.69 kernel  ninit:0: transport provider:
   T_COTS_ORD
00:00000:00001:2003/03/03 16:50:30.73 server  Recovering database 'userdb1'.
00:00000:00001:2003/03/03 16:50:30.75 server  Checking external objects.
00:00000:00001:2003/03/03 16:50:30.76 server  The transaction log in the
database
   'userdb1' will use I/O size of 4 Kb.
00:00000:00001:2003/03/03 16:50:30.78 server  Database 'userdb1' is now
online.
00:00000:00001:2003/03/03 16:50:30.78 server  Recovery complete.
```

```
00:00000:00001:2003/03/03 16:50:30.78 server  SQL Server's default unicode
sort
   order is 'binary'.
00:00000:00001:2003/03/03 16:50:30.78 server  SQL Server's default sort order
is:
00:00000:00001:2003/03/03 16:50:30.78 server    'bin_iso_1' (ID = 50)
00:00000:00001:2003/03/03 16:50:30.78 server  on top of default character set:
00:00000:00001:2003/03/03 16:50:30.78 server    'iso_1' (ID = 1).
00:00000:00001:2003/03/03 16:50:30.78 server  Master device size: 225
megabytes,
   or 115200 virtual pages. (A virtual page is 2048 bytes.)
00:00000:00011:2003/03/03 16:50:30.79 kernel  nconnect: t_rcvconnect,
   An event requires attention
```

### 8.1.4.2. Start up of Sybase Backup Server

Starting the Sybase backup server is completely optional in this environment. For completeness, we started the backup server using the replicated shell script.

```
venus% ./RUN_sri_back &
[2] 735
venus% Backup Server/12.5.0.3/EBF 10688 IR/P/Sun_svr4/OS
   5.8/rel125x/2251/32-bit/OPT/Sat Jan  4 05:05:00 2003
Confidential property of Sybase, Inc.
Copyright 1987, 2003
Sybase, Inc.  All rights reserved.
Unpublished rights reserved under U.S. copyright laws.

This software contains confidential and trade secret information of Sybase,
Inc.
Use, duplication or disclosure of the software and documentation by the  U.S.
Government is subject to restrictions set forth in a license agreement between
the Government and Sybase, Inc. or other written agreement specifying the
Government's rights to use the software and any applicable FAR provisions,
for example, FAR 52.227-19.

Sybase, Inc. One Sybase Drive, Dublin, CA 94568, USA

Logging Backup Server messages in file '/sybase/ase125/ASE-
12_5/install/sri_back.log'
```

### 8.1.4.3. Starting Sybase Monitor Server

We started the Monitor server:

```
venus% ./RUN_sri_mon &
[3] 738
venus% Monitor Server/12.5.0.3/EBF 10689 IR/64bit/1735/P/Sun_svr4/OS
5.8/OPT/Sat
   Jan  4 07:03:10 2003

Confidential property of Sybase, Inc.
Copyright 1993, 2003
Sybase, Inc.  All rights reserved.
Unpublished rights reserved under U.S. copyright laws.
```

```
This software contains confidential and trade secret information of Sybase,
Inc.
Use, duplication or disclosure of the software and documentation by the U.S.
Government is subject to restrictions set forth in a license agreement between
the Government and Sybase, Inc. or other written agreement specifying the
Government's rights to use the software and any applicable FAR provisions,
for example, FAR 52.227-19.
Sybase, Inc. One Sybase Drive, Dublin, CA 94568, USA


Configuration Information:

Server name: sri
Server version: Adaptive Server Enterprise/12.5.0.3/EBF 10689
   IR/P/Sun_svr4/OS 5.8/rel12503/1915/64-bit/FBO/Thu Jan 23 16:05:19 2003
Server shared memory size (bytes)  = 49676288
Maximum number of server engines   = 1
Number of event buffers configured = 100
Procedure mon_rpc_attach results: 0x200000000, 0x2f60000, 4441, 0x2001a8000

Monitor Server name: sri_mon
Size of memory blocks used for summarizing events = 32 KB
Maximum number of memory blocks per summary = 32
Maximum number of summaries per client connection = 15
Maximum number of client connections = 5
Maximum number of filtering criteria = 25
Maximum value permitted for pid, dbid and engine_id filters = 1023
Initial amount of memory required by the monitor server = 822 KB

Initialization is over. Ready to accept connections
```

### 8.1.5. Check for Data Availability

After the successful ASE server startup, check carefully whether all the databases are recovered and that you can continue to access the ASE server and all the databases. In our test setup, we connected to the ASE server using the "isql" utility and checked that the mirrored data had actually been available on the target location and no data corruption had occurred. The following output on our test environment shows that the new database has all the same records available as the source filer. It is evident that we created a new table called "class" and added several rows to the table "title," and this data is reflected on the target location.

```
venus% isql -Usa -P
1> use userdb1
2> go
1> select count(*) from title
2> go
-----------
     1048576

(1 row affected)
1> select count(*) from class
2> go

 -----------
```

```
      524288

(1 row affected)
1> select count(*) from titlec
2> go


 -----------
      524288

(1 row affected)
1>
```

The above output shows that all the data has been mirrored properly to the target location and ASE is running successfully at a secondary site.

## 8.2. sp_resetstatus Script as Provided by Sybase ASE Product Manual

If the file permission and file ownerships are properly configured for "sybase" account user on the target machine directories, the start of ASE should be successful depending on the system configuration and availability of essential system resources such as memory and network.

If you encounter an error message saying that a Sybase device could not be recovered or accessible, check the file permission and make files accessible to "sybase" user. If a user database is marked as suspect, it can be reset using the following script that is available on the Sybase external site. Note that this script is not officially supported by either Sybase, Inc. or Network Appliance, Inc. It is listed below:

```
CREATE PROC sp_resetstatus @dbname varchar(30) AS
DECLARE @msg varchar(80)
IF @@trancount > 0
    BEGIN
      PRINT "Can't run sp_resetstatus from within a transaction."
      RETURN (1)
    END
IF suser_id() != 1
    BEGIN
     SELECT @msg =  "You must be the System Administrator (SA)"
     SELECT @msg = @msg + " to execute this procedure."
     PRINT @msg
     RETURN (1)
    END
IF (SELECT COUNT(*) FROM master..sysdatabases
     WHERE name = @dbname) != 1
    BEGIN
     SELECT @msg = "Database '" + @dbname + "' does not exist!"
     PRINT @msg
     RETURN (1)
    END
IF (SELECT COUNT(*) FROM master..sysdatabases
     WHERE name = @dbname AND status & 256 = 256) != 1
   BEGIN
      PRINT "sp_resetstatus may only be run on suspect databases."
      RETURN (1)
    END
```

```
BEGIN TRAN
   UPDATE master..sysdatabases SET status = status - 320
     WHERE name = @dbname
   IF @@error != 0 OR @@rowcount != 1
     ROLLBACK TRAN
   ELSE
       BEGIN
         COMMIT TRAN
         SELECT @msg = "Database '" + @dbname + "' status reset!"
         PRINT @msg
         PRINT " "
         PRINT "WARNING: You must reboot Adaptive Server prior to  "
         PRINT "          accessing this database!"
         PRINT " "
       END
```

To create the stored procedure sp_resetstatus using the "isql" utility, run this command using the master database. ASE server needs to restarted to effect the changes made to the stored procedure. After the ASE is restarted, it can be executed with the "sp_resetstatus dabasename" command using the isql utility. An example is given below:

```
1> sp_resetstatus userdb1
2> go
Database 'userdb1' status reset!

WARNING: You must reboot Adaptive Server prior to
          accessing this database!
1> shutdown
2> go
```

The replicated volume can be used for reporting purposes without interrupting the production environment. To achieve this requirement, use SnapMirror to replicate the data to a target location and configure the target location on a different server. After the reporting or testing is over, shut down the ASE server and other Sybase servers on the target location. The target location is again ready for data replication from the source. Note that ASE may not be started using the read-only option.

## 9. Enabling NVFAIL on the NetApp Filer to Support Databases

When storing database data on a NetApp filer, it is suggested you enable the "nvfail" feature of the Data ONTAP operating system software. The nvfail feature provides support for special error processing. To enable this feature, use the following command syntax:

```
# vol options [VOL_NAME] nvfail on
```

where VOL_NAME is the name of the filer volume the nvfail feature is to be enabled for.

When the nvfail option is turned on the NetApp filer will store appropriate error message information in the /etc/messages file (stored on the root volume of the NetApp filer) in the event of a system failure that may have an effect on database server data integrity. The system administrator can then learn of these occurrences by examining the message logs on the filer or by e-mail if the autosupport e-mail notification feature of the filer is enabled.

Furthermore, when the nvfail option is enabled, additional status checking is performed to verify that

the NVRAM is in a "valid" state when the filer goes through its initialization sequence at boot time. (This check will be performed regardless of whether the filer was manually shut down by a system administrator or was inadvertently shut down because of a system crash, power failure, etc.) If the filer was inadvertently shut down because an NVRAM failure occurred, status checking will indicate the NVRAM is in an "invalid" state and the contents of the NVRAM will be examined. If the contents of the NVRAM are found to be invalid, an error message will be displayed on the system console and diagnostic information will be written to the NetApp filer log file. Invalid NVRAM data will also cause all DAFS connections to fail with "Stale File Handle" errors—this, in turn, will cause the Sybase database to hang or shut down and the Sybase DBA will need to check that the state of the database is consistent and valid.

If desired, a second feature that renames certain files that the system administrator or DBA may not want to be made accessible to the network until after they have been carefully examined can be used to provide additional protection. This feature is controlled by the presence or absence of the /etc/nvfail_rename file (stored on the root volume of the NetApp filer). The format of the /etc/nvfail_rename file is simply the name of a file found on the filer, one file name per line; if this file exists, each file listed in it is renamed by having the string ".nvfail" appended to it. Note: NVRAM is designed to work without any failure. If users are really concerned about a single point of hardware failure, they must consider enabling the nvfail feature or use the Cluster Fail-Over (CFO) solution from Network Appliance Inc. To ensure that a DBA becomes aware of an NVRAM failure regardless of any user attempts to access any database stored on a NetApp filer, you may want to create a /etc/nvfail_rename file that contains the names of each file on the filer that is being used by a Sybase database.

## 10. Information about ASE 12.0 and ASE 11.9.2

The information provided in this paper can be used while working with ASE 12.0 and ASE 11.9.2. This paper is applicable for both 32-bit and 64-bit ASE products. The procedure is same on other flavors of UNIX such as IBM AIX, HP/UX, and SGI IRIX. The only change required is mount command syntax. Refer to the particular OS vendor's manuals for more details.

## 11. Disaster with Power Failure

We simulated a disaster scenario by pulling the power plug of both the filer and the Sun server. The disaster recovery steps involved during the database activities are listed in this section. The following example shows that we created a Sybase physical device and created a database. The power supply was pulled during updating of the table.

```
1> disk init name="userdev3",
2> physname = "/sybase/data/userdev3",
3> size='5g'
4> go
00:00000:00012:2003/03/04 13:39:47.96 kernel  Initializing virtual device 5,
    '/sybase/data/userdev3' with dsync 'on'.
00:00000:00012:2003/03/04 13:39:47.96 kernel  Virtual device 5 started using
    asynchronous i/o.
00:00000:00012:2003/03/04 13:39:47.96 kernel  Initializing device
    /sybase/data/userdev3 from offset 0 with zeros.
00:00000:00012:2003/03/04 13:42:15.28 kernel  Finished initialization.
```

```
1> disk init name="logdev3",
2> physname = "/sybase/data/logdev3",
3> size='3g'
4> go
00:00000:00012:2003/03/04 13:43:39.96 kernel  Initializing virtual device 6,
    '/sybase/data/logdev3' with dsync 'on'.
00:00000:00012:2003/03/04 13:43:39.96 kernel  Virtual device 6 started using
    asynchronous i/o.
00:00000:00012:2003/03/04 13:43:39.96 kernel  Initializing device
/sybase/data/
    logdev3 from offset 0 with zeros.
00:00000:00012:2003/03/04 13:45:27.13 kernel  Finished initialization.
1>
1> create database userdb3 on userdev3=3000 log on logdev3=2400
2> go
CREATE DATABASE: allocating 768000 logical pages (3000.0 megabytes) on disk
'userdev3'.
CREATE DATABASE: allocating 614400 logical pages (2400.0 megabytes) on disk
'logdev3'.
1>
filerB> snapmirror resync -S 10.32.90.15:sybsource sybtarget
The resync base snapshot will be: filerB(0033587173)_sybtarget.2
These newer snapshots will be deleted from the destination:
    hourly.0
    filerB(0033587173)_sybmirr.1
    hourly.1
    nightly.0
    hourly.2
    hourly.3
These older snapshots have already been deleted from the source
and will be deleted from the destination:
    filerB(0033587173)_sybtarget.1
Are you sure you want to resync the volume?  y

and will be deleted from the destination:
    filerB(0033587173)_sybtarget.1
Are you sure you want to resync the volume? y
Tue Mar  4 13:32:50 PST [rc:notice]: snapmirror: resynching volume sybtarget
to
    10.32.90.15:sybsource using filerB(0033587173)_sybtarget.2 as the base
snapshot.
Volume sybtarget will be briefly unavailable before coming back online.
Tue Mar  4 13:32:50 PST [config_async_0:notice]: CIFS - disabled for volume
sybtarget
Volume sybtarget is now restricted.
Tue Mar  4 13:32:51 PST [config_async_0:notice]: Reverting volume sybtarget to
    a previous snapshot.
Tue Mar  4 13:32:51 PST [license.db.migrate.vol.readonly:warning]: Skipping
volume
    sybtarget migration because it is read-only
Volume 'sybtarget' is now online.
Volume sybtarget: revert successful.
Revert to resync base snapshot was successful.
Transfer started.
Monitor progress with 'snapmirror status' or the snapmirror log.
```

```
filerB>
```

Create a table called "info" and populate that table. While populating the table, assume that the ASE UNIX server experienced the power failure or a disaster, the ASE server crashes, and the SnapMirror process gets aborted. Now we will try to recover the database at the target site. On the target filer, we broke SnapMirror and made the target volume writeable. As mentioned earlier, we will change the file ownership and permissions for "sybase" user account. If you encounter stale NFS handle, unmount and mount the file system.

After mounting the target volume, we changed the file ownership to be accessed by "sybase" user on the disaster site. The following output shows that all the databases were recovered successfully even after the crash at the source site.

```
00:00000:00001:2003/03/04 15:15:55.14 server  Recovering database 'userdb1'.
00:00000:00001:2003/03/04 15:15:55.17 server  Checking external objects.
00:00000:00001:2003/03/04 15:15:55.17 server  The transaction log in the
database
    'userdb1' will use I/O size of 4 Kb.
00:00000:00001:2003/03/04 15:15:55.19 server  Database 'userdb1' is now
online.
00:00000:00001:2003/03/04 15:15:55.23 server  Recovering database 'userdb3'.
00:00000:00001:2003/03/04 15:15:55.26 server  Checking external objects.
00:00000:00001:2003/03/04 15:15:55.26 server  The transaction log in the
database
    'userdb3' will use I/O size of 4 Kb.
00:00000:00001:2003/03/04 15:15:55.28 server  Database 'userdb3' is now
online.
00:00000:00001:2003/03/04 15:15:55.29 server  Recovery complete.

We checked the number of records on the new table where the data was getting
inserted
at the source site.  The output is shown below:

1> use userdb3
2> go
1> select count(*) from info
2> go

       16384

(1 row affected)
1>
```

Refer to SnapMirror documentation to learn in detail how to configure the mirror data with minimal or no loss. The documentation is available at our external NOW site

## 12. Summary of Steps Involved with SnapMirror (Volume or qtree Level)

1. Make sure you have purchased a SnapMirror license.

2. Add the SnapMirror license by using the license add command.

3. On the source filer console, use the snapmirror.access command to specify the host names of filers that are allowed to copy data directly from the source filer. For example:

4.
5. `options snapmirror.access host=d_filerA`
6. Through the Data ONTAP AdminHost, create or edit the /etc/snapmirror.conf file on the destination filer to specify the volumes and qtrees to be copied and the schedule (minute hour day_of_month day_of_week) on which the destination is updated. For example, the following entry specifies Snapshot mirroring from volume 0 of s_filerA to volume 1 of d_filerA at a maximum of 5000Kb per second every 15 minutes, of every hour, of every day Monday through Friday.

   s_filerA:vol0 d_filerA:vol1 kbs=5000,restart=always 15 *
   * 1,2,3,4,5

7. On both source and target filers, use "snapmirror on" to enable mirroring.

8. Configure the mirroring destination carefully, depending on whether you are setting up SnapMirror volume or qtree replication. On the destination filer, use "vol restrict volname" command where volname is name of volume or qtree.

9. Use the SnapMirror initialize command to create an initial complete (baseline) copy of the source on the destination and start the mirroring process.
   *SnapMirror volume replication example:*
   Invoking the following command line transfers a complete copy of the source volume (vol0 on filerA) to the destination volume (vol2 on filerB). The destination volume must be configured as restricted and read-only.

   ```
   snapmirror initialize -S filerA:vol0 filerB:vol2
   ```

   *SnapMirror qtree replication example:*
   Invoking the following command line creates a destination qtree (qtree4 on vol1on filerB) and transfers a complete copy of the qtree source (qtree4 on vol1 on filerA) to that destination qtree. The volume in which the destination qtree is created must be online and writeable.

   ```
   snapmirror initialize -S filerA:/vol/vol1/qtree4
   filerB:/vol/vol1/qtree4
   ```

10. In the event that the SnapMirror source volume or qtree is disabled, you can use the SnapMirror break command to make the destination volume or qtree writeable and able to provide continuous file and data service to the clients who are no longer able to access the disabled source filer.

## 13. Caveats

Sybase has certified Network Appliance filers for storing Sybase Adaptive Server database files. Network Appliance, using certain sets of hardware and software options, has tested the configuration presented in this paper. Therefore, your experience may differ from that presented here. This paper has no intention of replacing any documents. If you find any information different from official documents, contents of those documents are to be considered as valid, or if you have any problems with the information provided in this technical report, please contact the author at Network Appliance, Inc.

# 14. References


- TR-3014: Multiprotocol Data Access: NFS, CIFS, and HTTP
- SnapMirror and SnapRestore: Advances in Snapshot Technology
- Sybase ASE 12.5 System Administrators Guide
- TR-3166: Integrating Quiesce Database
- SnapMirror Operational Guide


[1] Mount root volumes of both filers to access the SnapMirror control files. File names are in lowercase.

NetworkAppliance®
The evolution of storage.™

**Network Appliance, Inc.**
495 East Java Drive
Sunnyvale, CA 94089
www.netapp.com